Arab Academy for Science &Technology and Maritime Transport (AASTMT)
College of Computing and Information Technology (CCIT)
Computing Alg. CS312
Dr. Manal Helal
Eng. Mohamed Moheeb

Section 1 – 2nd-of March 2014

Q1. New World puzzle There are four people who want to cross a rickety bridge; they all begin on the same side. You have 17 minutes to get them all across to the other side. It is night, and they have one flashlight. A maximum of two people can cross the bridge at one time. Any party that crosses, either one or two people, must have the flashlight with them. The flashlight must be walked back and forth; it cannot be thrown, for example. Person 1 takes 1 minute to cross the bridge, person 2 takes 2 minutes, person 3 takes 5 minutes, and person 4 takes 10 minutes. A pair must walk together at the rate of the slower person's pace. (Note: According to a rumor on the Internet, interviewers at a well-known software company located near Seattle have given this problem to interviewees.)

Q2. Consider the algorithm for the sorting problem that sorts an array by counting, for each of its elements, the number of smaller elements and then uses this information to put the element in its appropriate position in the sorted array:

ALGORITHM *ComparisonCountingSort*(A[0..n − 1])

//Sorts an array by comparison counting

//Input: Array A[0..n − 1] of orderable values

//Output: Array S[0..n − 1] of A's elements sorted

// in nondecreasing order
for i ← 0 to n − 1 do

```
        Count[i]←0
    for i ← 0 to n − 2 do
        for j ← i + 1 to n − 1 do
            if A[i] < A[j]
                Count[j]←Count[j]+1
            else
                Count[i] ← Count[i] + 1
```

```
        for i ← 0 to n − 1 do
                S[Count[i]]←A[i]
        return S
```

a. Apply this algorithm to sorting the list:
60, 35, 81, 98, 14, 47.

b. Is this algorithm stable?

c. Is it in-place?

Q3. Describe how one can implement each of the following operations on an array so that the time it takes does not depend on the array's size n.

a. Delete the ith element of an array($1 \le i \le n$).
b. Delete the ith element of a sorted array (the remaining array has to stay sorted, of course)